J. Symbolic Computation (1986) **2**, 217–236

Computer Algebra: Past and Future†

B. F. CAVINESS

Department of Computer and Information Sciences,
University of Delaware, Newark, Delaware 19716, *U.S.A.*

(*Received* 20 *March* 1986)

This survey reviews some of the important accomplishments in computer algebra since 1966 and indicates some directions for future research and development.

1. Introduction

In this survey, I wish to recount some of the important accomplishments in computer algebra and to explore some of the significant directions for future research and development. I will take 1966 as my beginning point. This seems appropriate for a variety of reasons: (i) 1966 was the year of the first two conferences on this subject, namely, the ACM Symposium on Symbolic and Algebraic Manipulation held in Washington, D.C., 29–31 March 1966 (Floyd, 1966) and the IFIP working conference on Symbol Manipulation Languages and Techniques held in Pisa, 5–9 September 1966. (ii) The 1966–85 period is the span of my own professional career in computer algebra, for it was at the meeting in Washington that I learned about the results of Dan Richardson (1968) on the unsolvability of both the zero identity problem and integration in finite terms. This work stimulated me to work on some problems that eventually led to my PhD dissertation.

Further, I wish to focus on the recent past. Prior to 1966 much work was done that is relevant and important for current day computer algebra; indeed, too much research to survey in the time and space that we have here. Before turning to the last 20 years, I would like to make one observation about earlier research. Previous to, say 1850, the work of a mathematician was much different from today. In particular, computation played a much larger role in problem solving and was sometimes instrumental in establishing one's reputation.

Take Gauss as a notable example. As reported by E. T. Bell's (1937) colourful history, *The Men of Mathematics*, Gauss began his career with the study of number theory and in 1801 published his treatise *Disquisitiones Arithmeticae*, which was immediately recognised as a masterpiece by competent judges, but did not bring widespread fame. According to Bell, to establish his reputation as a mathematician of first rank, Gauss apparently felt it necessary to temporarily abandon his first love, number theory, and to bring his formidable calculating talents to the task of determining the orbit of Ceres, an insignificant minor planet that had been discovered in 1800. When it reappeared the next year in the exact spot that Gauss' calculations had predicted, his reputation was firmly

† An earlier draft of this paper was presented as an invited contribution at the EUROCAL 85 Conference, 1–3 April 1985, Linz (Springer-Verlag Lecture Notes in Computer Science, Vol. 203, pp. 1–18).

0747–7171/86/030217+20 $03.00/0

established. Bell laments, "Ceres was a disaster for mathematics. To understand why she was taken by such devastating seriousness by Gauss we must remember that the colossal figure of Newton—dead for more than 70 years—still overshadowed mathematics in 1801. The 'great' mathematicians of the time were those who, like Laplace, toiled to complete the Newtonian edifice of celestial mechanics. Mathematics was still confused with mathematical physics—such as it was then—and mathematical astronomy. The vision of mathematics as an autonomous science which Archimedes saw in the third century before Christ had been lost sight of in the blaze of Newton's splendour, and it was not until the youthful Gauss again caught the vision that mathematics was acknowledged as a science whose first duty is to itself. But that insignificant clod of dirt, the minor planet Ceres, seduced his unparalleled intellect when he was 24 years of age, just as he was getting well into his stride in those untravelled wildernesses which were to become the empire of modern mathematics."

You may or may not agree with Bell's assessment of Gauss. He (Gauss) not only calculated the orbit of Ceres, but introduced important new computational methods in the process, e.g. least square fittings. The main point that I wish to make is that computation was a major part of a mathematician's work at that time.

However, by the end of the nineteenth century, with much of the credit due to Gauss, mathematics had changed substantially; it had become much more qualitative and less computational. Mathematicians concerned themselves more with questions about the structure of mathematics than they did with the computation of solutions to particular problems. There were many reasons for this change, but one reason seems to be that it was becoming increasingly difficult to solve significant problems by computational means.

With the aid of computers, the solution of problems via computation has once again become feasible, and there are many points of contact between modern day computer algebra and earlier mathematics. In particular, mathematicians of the quantitative era were superbly skilled at computation and their papers contain many valuable insights for us. This historical heritage from mathematics gives an important depth and richness to computer algebra. I wish to acknowledge the importance of this heritage before moving on to the more recent past.

2. Some Important Accomplishments of the 1966–1985 Period

Before beginning work on this paper, my impression as a member of the computer algebra community during this period was that our progress had been slow and that the accomplishments of lasting value were few. But as I began to review the literature and tally the results, I decided that the record is not bad at all—in fact, it is pretty impressive. Perhaps my initial negative feelings are the result of the slow progress of my own work!

In selecting research to single out as important and distinctive, I became acutely aware of my limited vision of the world. One naturally tends to think that the problems on which he or she works are the most important. I have consulted with a number of persons to try to eliminate as much of my personal bias as possible, but I am solely responsible for final choices and any errors are due entirely to me.

Computational group theory and computational number theory will not be included in this survey. A survey on computational group theory is planned for a future issue of this journal.

In the following I distinguish between practically fast algorithms and theoretical results. The former includes algorithms that have proved useful in practice, even though they may

not have the best asymptotic computing-time bound. The algorithms discussed under theoretical results have solved some problem of a theoretical or mathematical nature. They may or may not be useful in practice. In some cases they have not been implemented.

2.1. PRACTICALLY FAST ALGORITHMS FOR COMPUTATION OF POLYNOMIAL GCD'S AND FACTORISATION

It seems appropriate to begin any list of notable accomplishments in computer algebra with the advances that have been made in practically fast algorithms for the computation of polynomial gcd's and polynomial factorisation, for without these advances, computer algebra as we know it today simply would not exist. The application of modular and p-adic arithmetic along with attention to sparseness has made these advances possible. We will be concerned primarily with polynomials in $Z[x_1, \ldots, x_r]$, i.e. r-variate ($r > 0$) polynomials over the integers.

The first instance, known to me, of an implementation of a gcd algorithm for multivariate polynomials is of the Euclidean algorithm in the PMS system of Williams (1962) on an IBM 650. Apparently because of storage limitations, he was able to carry out limited computations and only hinted at some of the problems in using Euclid's algorithm. A variant of Euclid's algorithm, called the primitive PRS (polynomial remainder sequence) algorithm, was used in the ALPAK system and in the 1964 paper by Brown *et al.*, the coefficient growth that occurs is discussed. However, the substantial coefficient growth they observed in practice did not imply an exponential computing time as they suggested (cf. Collins, 1966; Heindel, 1971; Loos, 1982).

The first major advance in computing gcd's was Collins (1967) subresultant PRS algorithm, which controlled coefficient growth without requiring that each polynomial in the remainder sequence be primitive. In Loos (1982), an improved subresultant PRS algorithm is presented, and the theory behind this and related algorithms is nicely described. Also see Brown & Traub (1971) and Brown (1976) for similar material. The improved subresultant PRS algorithm appears to be the best non-modular gcd algorithm.

Next, Brown & Collins (1972), "working independently but with some communication" (Brown, 1971), discovered the modular algorithm that employs reductions modulo a prime and reconstructions via the Chinese remainder algorithm (CRA). Knuth also worked out the details of this algorithm from ideas suggested by Brown and Collins and published a first sketch of it in 1969. Brown's paper contains important improvements, not in Knuth's description, for multivariate polynomials and for the computation of co-factors. The modular gcd algorithm has a maximum computing time $= O(n^{2r+1}L(d)^2)$ for polynomials in $Z[x_1, \ldots, x_r]$, where n is the maximum degree in any variable and $L(d)$ is the maximum length of the semi-norms of the two polynomials, i.e. the sum of the absolute values of the coefficients.

The next step in the improvement of gcd algorithms was sparked by a paper of Zassenhaus (1969) in which he published a quadratic version of the Hensel lifting algorithm in p-adic fields and generally brought these important techniques to the attention of the computer algebra research community. Moses & Yun (1973) used modular homomorphisms and Hensel p-adic lifting in their gcd algorithm to take advantage of sparseness, something that the earlier CRA-based algorithms did not do.

Zippel in his 1979 thesis (1979*a*, *b*) developed probabilistic algorithms for sparse polynomials that used an important new technique for preserving sparsity of a

multivariate gcd during Brown's interpolation scheme. His algorithm is currently the default gcd algorithm used in MACSYMA. In favourable cases it can compute gcd's of two sparse polynomials in ten variables with single-digit coefficients in less than nine seconds on a PDP-10.

Now we turn to the problem of factoring polynomials. The initial work on polynomial factorisation algorithms by Jordon *et al.* (1966) and Johnson (1986) was based on Kronecker's algorithm as given by van der Waerden (1953). This algorithm requires exponential computing time and is not useful in practice.

In 1967 Berlekamp published an algorithm for factoring polynomials over GF(p) that was to become the basis for many improvements. This along with Zassenhaus' (1969) paper on Hensel lifting has led to major theoretical and practical advances.

Musser (1971, 1976), in two excellently written papers, generalised the Hensel lemma to obtain multivariate factorisation algorithms. Wang & Rothschild (1975) published similar algorithms. Wang (1977, 1978, 1979) has contributed additional practical improvements to the factorisation algorithms and his EEZ algorithm, a variant of the Berlekamp–Hensel algorithm, is the one that is still used today for multivariate factorisation in MACSYMA.

Although all these algorithms, in the worst case, require exponential time in the degree of the polynomial, they seem to work reasonably well in practice. However, for some polynomials the Berlekamp–Hensel algorithm can take a long time (Kaltofen *et al.*, 1983).

Collins' (1979) paper shows, subject to some conjectures, that the average computing time of a particular formulation of the Berlekamp–Hensel scheme applied to a primitive univariate integral polynomial is dominated by a polynomial function of its degree. Hence, he provides an argument that lends theoretical evidence to the experimental results that show that these algorithms are often reasonably fast. The complete analysis of the average computing time of the Berlekamp–Hensel algorithm remains an open problem.

There is a large literature on many aspects of factoring polynomials and computing gcd's. We have mentioned some of the highlights. For comprehensive surveys, see Kaltofen's (1982*a*) paper on factorisation, Loos' (1982) article on the theoretical underpinnings of gcd algorithms and Lauer's (1982) study on computing by homomorphic images.

Another important problem is that of determining a numerical approximation to the roots of a univariate polynomial in the following sense. Given a small rational number $\varepsilon > 0$ and $p(x)$ in $Z[x]$, determine disjoint intervals on the real line of length less than ε, each containing exactly one real zero of p, and determine disjoint rectangles in the complex plane with sides less than ε, each containing exactly one complex root of p, and together the isolating intervals and rectangles contain all the roots of p. To guarantee that these requirements are met requires an algebraic approach that is different, and usually computationally more expensive, than the classical numerical methods for approximating roots without guaranteed accuracy.

Collins & Loos (1983) survey methods for isolating real roots of polynomials. The modified Uspensky algorithm (Collins & Akritas, 1976) seems to be the best algorithm for this problem. After isolating the zeros, the intervals must be refined using a method such as bisection or Newton's method. Pinkert (1976) has given methods for isolating and refining complex roots. Pan (1984) gives fast algorithms for these problems using both sequential and parallel computational models.

2.2. THEORETICAL RESULTS IN GCD COMPUTATION AND POLYNOMIAL FACTORISATION

Although we have concentrated so far on algorithms for polynomials in $Z[x_1, \ldots, x_r]$, I would like to begin this section by noting some results on the analysis of Euclid's algorithm applied to rational and Gaussian integers. Finding a tight bound on the number of divisions required by the Euclidean algorithm over Z is an easy problem, the solution of which follows from an 1845 result of Lamé (see Knuth, 1969, section 4.5.3). In 1843 Dupré obtained the corresponding result for the least remainder version of the Euclidean algorithm over Z.

Let τ_n be the average number of division steps required by the Euclidean algorithm when applied to integers u and v with $v = n$ and u relatively prime to n. In 1969 Knuth stated that: "In view of the historical importance of Euclid's method, it seems fair to state that a determination of the asymptotic behaviour of τ_n is the most important problem in the analysis of algorithms which is still unsolved. The world's most famous algorithm deserves a complete analysis!"

The analysis of the Euclidean algorithm over Z was completed between 1969 and 1975 in a series of papers by Dixon (1970), Heilbronn (1969), Porter (1975) and Collins (1974). Porter confirmed Knuth's earlier estimate of τ_n by showing that

$$\tau_n = \frac{12 \ln 2}{\pi^2} \ln n + C + O(n^{-1/6+\varepsilon}),$$

where C is approximately 1.467 . . ., and Collins completed the analysis by showing that the average computing time is

$$O\left(\left[1 + \log\left(\frac{\max(u, v)}{\gcd(u, v)}\right)\right] \log \min(u, v)\right).$$

In the early 1970s I was surprised to learn that apparently no tight bound had been obtained for the number of divisions required by the Euclidean algorithm when applied to elements of $Z[i]$, the Gaussian integers. It was easy to find a plausible conjecture for this case, but the proof eluded me. I showed the problem to several persons; it turned out to be surprisingly slippery. Rolletschek finally resolved the problem in 1983 by showing that the number of divisions required by the Euclidean algorithm applied to u, v in $Z[i]$ with $0 \leqslant |v| \leqslant |u| \leqslant N$ is at most $1.053 \log_2 N + 2$ (this is a simplified statement of his results).

Now let us turn to additional results for polynomials. As I have just recounted, substantial progress on practical algorithms for computing gcd's and factoring polynomials was made starting in the 1960s. However, the problem of finding an algorithm for factoring univariate polynomials with integer coefficients with a worst case computing time bounded by a polynomial function of the degree and the length of the coefficients remained unsolved. By the mid-1970s, it was considered a major open problem in computer algebra. (There was no corresponding open problem for gcd computation since the early algorithms were known to be polynomial time (see Heindel, 1971, p. 537).)

In 1982 this problem was solved for both the univariate and multivariate cases by several persons. Lenstra *et al.* (1982) solved the problem for the univariate case. Kaltofen (1982*b*, *c*, *d*, 1985) showed how to reduce the problem of multivariate factorisation in r variables over Z ($r > 1$ and fixed) to univariate factorisation. His reduction, with the LLL algorithm, showed that multivariate polynomials can be factored in polynomial time. But these algorithms are still exponential in the number of variables. This is the best that can

be expected in general since sparse polynomials can have dense factors. In 1982 Chistov & Grigor'ev independently obtained results similar to Kaltofen (1982*c*) and extended the results discussed above to more general coefficient domains. The problem of finding a polynomial-time algorithm for factoring polynomials over algebraic extensions of Q has been solved by Chistov & Grigor'ev (1982), Landau (1985) and Lenstra (1983). Lenstra (1984) gives a short, interesting survey of recent results on polynomial factorisation and discusses the relationship of polynomial factorisation to other problems like testing a complex number for algebracity and breaking public-key cryptosystems.

In 1983 von zur Gathen presented a probabilistic algorithm that factors sparse multivariate polynomials over Z in expected polynomial time as a function of the number of variables, length of the coefficients, total degree and the number of monomials in the input and output, i.e. factors in expected polynomial time, sparse polynomials whose factors are sparse.

A problem related to factorisation is that of determining the roots of a polynomial. Also in 1983 Landau & Miller found a polynomial time algorithm for determining if the roots of a polynomial can be expressed in terms of radicals thus solving another long standing problem.

Slisenko (1981), in an extensive survey on computational complexity, discusses complexity results for a number of algebraic problems. His paper contains an excellent bibliography.

2.3. ALGORITHMIC POLYNOMIAL IDEAL THEORY

There are many important computational problems that can be studied profitably in the framework of polynomial ideals. Given a set $P = \{p_1, \ldots, p_n\}$ of polynomials in $K = Q[x_1, \ldots, x_r]$, Q the field of rational numbers, the ideal generated by P, denoted ideal (P), is the set

$$\left\{ \sum_{i=1}^{n} q_i p_i | q_i \in Q[x_1, \ldots, x_r] \right\}.$$

Using the unifying concept of a Gröbner basis (Buchberger, 1965), a variety of problems connected with systems of polynomial equations can be solved, including exact solution of a system of polynomial equations, computation in the residue class ring modulo an ideal, and canonical simplification with respect to algebraic relations.

Hermann (1926) was an early investigator of constructive methods in this area and her paper gives an algorithm for determining if a polynomial $q \in K$ is in an ideal (P). In the succeeding years this and related questions have been studied in both the mathematics and computer algebra research communities. I will concentrate on the literature that is concerned with finding efficient algorithms for polynomial ideal theory.

Much of the recent work can be traced to Buchberger's Ph.D. thesis (1965) in which he gave an algorithm for constructing a canonical basis, which he called a Gröbner basis, for an ideal. That is, given a finite set P of multivariate polynomials over a field, find a finite set P' such that ideal (P) = ideal (P') and the normal form of a polynomial modulo P becomes a canonical form modulo P'. See Buchberger (1985*a*) for the definition of the reduction process and a more complete survey of recent accomplishments and of applications of Gröbner bases.

There are many similarities between Buchberger's algorithm and the Knuth–Bendix completion procedure. Some of these are discussed by Buchberger & Loos (1983) and

Kandri-Rody & Kapur (1983). Algorithms for constructing a Gröbner basis for polynomials over coefficient domains other than a field are discussed by Lauer (1976), Zacharias (1978), Schaller (1979), Buchberger (1983*b*) and Kandri-Rody & Kapur (1984*a*, *b*).

In 1964 Hironaka defined the concept of a standard basis, which is essentially the same as Buchberger's Gröbner basis, but Hironaka did not give a constructive method for computing such a basis. The reduction process for reducing a polynomial to its normal form modulo an ideal that is described in Buchberger's paper is often called the division algorithm in the mathematical literature because the process reduces to calculating the remainder of the polynomial if the ideal basis consists of only one element.

The complexity of the Buchberger algorithm has been studied in several papers including Buchberger (1979, 1983*a*), Lazard (1983), Mora & Möller (1983) and Winkler (1984). In the latter, Winkler shows that, for $r = 3$, the degree of every polynomial that is constructed during the execution of Buchberger's algorithm applied to a set of polynomials P to construct a Gröbner basis P' is at most $(8D+1)2^d$, where D and d are the maximal and minimal degrees of the polynomials in P. The degree of a term $x_1^{n_1}x_2^{n_2}x_3^{n_3}$ is $n_1+n_2+n_3$.

In Mayr & Meyer (1982) a lower bound is given on the computing time for this problem. They show, for $r = 1$, that the problem of determining if a polynomial p belongs to the ideal (P) requires space exceeding $2^{\varepsilon v}$ for infinitely many v, where ε is a positive constant and v is the sum of the lengths of the coefficients and exponents of $p, p_1, \ldots, p_n$. This is a fundamental result on the inherent complexity of the ideal membership problem.

Implementations of Buchberger's algorithm have been reported by Gebauer & Kredel (1984) and Winkler *et al.* (1985). The former is in SAC-2 and the latter in SAC-1.

The important problem of solving systems of polynomial equations can be approached by other methods than through the computation of a Gröbner basis. In this regard, the work of Chistov & Grigor'ev (1983*a*, *b*), Collins (1975), and Lazard (1981) should be noted. Chistov & Grigor'ev establish that this problem is not exponential in the number of variables.

2.4. DECISION PROCEDURES FOR LOGICAL THEORIES

Research in computer algebra has been connected to research on decision procedures for logical theories primarily through the work of Collins (1975) and his students on his quantifier elimination algorithm for the elementary theory of real closed fields. This work has at least partially motivated the work on efficient algorithms for various subtasks of this decision procedure including square free factorisation, real root isolation, computations with real algebraic numbers and construction of primitive elements for algebraic field extensions. Chistov & Grigor'ev (1984) have given a quantifier elimination algorithm for the theory of algebraically closed fields. Plans have been made for a future issue of this journal to be devoted to these topics.

Our presentation here is based on Collins (1983) and Arnon *et al.* (1984). The elementary theory of real closed fields is the first order theory with constants 0 and 1, function symbols $+$, $-$, $\times$, predicate symbols, $<$, $\leqslant$, $>$, $\geqslant$, $=$, the axioms of a commutative field, some axioms relating the order relation with the arithmetical operations and infinitely many axioms stating the existence of roots for certain equations.

The first decision procedure for this theory was given by Tarski (1951). His method, actually discovered in 1930, is a quantifier elimination method, i.e. to every formula F of

the theory a quantifier free formula F' is constructed such that F iff F'. Since the truth of a quantifier free formula is effectively decidable, this is a decision procedure.

Tarski's method as well as others by Seidenberg (1954), Cohen (1969) and an improved version of Tarski's method by Böge (1980) have computing times that are exponential in the number m of polynomials occurring in the formula F or the maximum degree n of these polynomials in a fixed number of variables, r. Collins' method depends only polynomially on m and n, but all methods are necessarily exponential in r as is implied by the results of Fischer & Rabin (1974).

An implementation of Collins' algorithm has been given by Arnon (1981). Various applications of the algorithm have been reported such as determining the termination of the term-rewriting system for group theory (Huet & Oppen, 1980), display of algebraic curves (Arnon, 1983), algebraic optimisation problems (Müller, 1978), and a problem posed by Delzell in connection with finding a solution of a certain kind to Hilbert's 17th problem for a form of degree four (Arnon, 1985).

The latter problem is to find conditions on the real numbers q, r and s such that the polynomial x^4+qx^2+rx+s is non-negative for all real x. In his paper, Arnon reports on a modification of Collins' algorithm that led to a solution as good as might be produced by a human. Delzell's problem was first solved using the original version of Collins' method. It required approximately 18 hours of CPU time on a VAX 11/780 and the answer was an unnecessarily complicated 80 lines long. After Arnon's changes the computation took 26 minutes. The input is the formula $(\forall\, x)(x^4+qx^2+rx+s \geqslant 0)$. Arnon's modified program produced the solution $\delta \geqslant 0$ and ($q \geqslant 0$ or $L<0$ or ($L=0$ and $r=0$)), where

$$\delta = 256s^3-128q^2s^2+144qr^2s+16q^4s-27r^4-4q^3r^2,$$

the discriminant of the input polynomial, and $L=8qs-9r^2-2q^3$.

This application suggests that Collins' algorithm, with suitable modifications, may be on the verge of being applicable for a number of important and difficult problems. It would be most interesting to see Arnon's work implemented on a fast machine and then systematically applied to various problems.

2.5. INTEGRATION AND SUMMATION IN FINITE TERMS

The publication of the Risch integration algorithm for the purely transcendental case (Risch, 1969*a*) and the general case, including algebraic functions (Risch, 1970), marked the solution of a problem that had intrigued mathematicians since Newton invented the calculus. The implementation of integration capabilities that employed this algorithm along with the more traditional heuristic techniques found in elementary calculus books played a major role in the early acceptance of computer algebra systems as useful and interesting tools. The first partial implementation of the Risch algorithm was by Moses (1967) in MACSYMA following an earlier 1961 implementation of heuristic techniques by Slagle. The paper by Harrington (1979) on the REDUCE integration package gives a good description of the interplay between heuristic and algorithmic methods that seems to be necessary to obtain an implementation with good human engineering factors.

Risch's 1970 paper omitted most of the details of the algorithm for integrating algebraic functions (although see his 1968 paper for additional algorithmic details), and the algorithm for integrating algebraic functions is the most complicated part of elementary function integration. Davenport (1981) in his Ph.D. thesis produced the first implementation of a substantial portion of the algebraic case and did a yeoman's job of

working out the substantial mathematical and algorithmic specifics that were required. Trager (1979), using a different approach from that of Davenport, has also made important contributions to algorithms for integrating algebraic functions.

Rothstein (1976) and Trager (1976) independently solved the problem of determining the minimal algebraic extension field needed to express the logarithmic part of an integral and gave algorithms for computing it.

Cherry (1983, 1985*a*, *b*) extended the Risch algorithm to a decision procedure for integrating transcendental elementary functions in terms of elementary functions plus logarithmic integrals or error functions. Cherry's results partially answer a 1969 conjecture by Moses that it was possible to extend the Risch algorithm in this way. Cherry's algorithms are based on an extension of the Liouville theorem for integration in finite terms given by Singer *et al.* (1985). Knowles (1986) has generalised both of Cherry's algorithms to allow arbitrary integrals in the integrand and enlarged the class of allowable exponentials in the integrand.

Some analogous work has also been done on algorithms for summation in finite terms. Gosper (1978) considers the problem of calculating the indefinite sum $S(n)$ determined by

$$\sum_{i=1}^{n} a_i = S(n) - S(0).$$

He gives an algorithm that given the a_i finds those $S(n)$ with the property that $S(n)/S(n-1)$ is a rational function. Gosper has implemented this algorithm in MACSYMA.

In his 1981 paper, Karr develops a theory of difference fields for first order linear difference equations that has certain analogues with differential fields and integration in finite terms. He shows how to solve arbitrary first-order, linear difference equation in a particular kind of difference field that is analogous in the finite difference case to the extension of the differential field $Q(x)$ by exponential and logarithmic functions. His 1985 paper discusses further theoretical aspects of the problem.

2.6. ALGORITHMS FOR SOLVING DIFFERENTIAL EQUATIONS IN CLOSED FORM

As with integration in finite terms, the problem of finding algorithms for computing closed form solutions of differential equations is a long standing one. The first breakthrough came with Kovacic's algorithm (1979) for solving second-order linear homogeneous differential equations. Soon thereafter, Singer (1981), in a significant paper, solved the general problem of finding an algorithm that determines Liouvillian solutions for nth order linear homogeneous differential equations with algebraic function coefficients, if such a solution exists. Thus their algorithms consider equations of the form

$$a_n(x)y^{(n)} + \ldots + a_1(x)y' + a_0(x)y = 0.$$

For Kovacic's algorithm $n = 2$ and each a_i is a rational function. Singer's algorithm is valid for any $n > 0$ with the a_i being algebraic functions. Both algorithms determine if a Liouvillian solution exists and, if so, finds it. A solution is a basis for the vector space that spans the solution space in which each element is Liouvillian, i.e. can be expressed in terms of algebraic functions, exponentials and integrals.

Saunders (1981) has implemented Kovacic's algorithm in MACSYMA and it seems to work reasonably well in practice. Singer's algorithm has not been implemented.

Prelle and Singer (Singer, 1977; Prelle, 1982; Prelle & Singer, 1983) have extended the

Liouville theory for another class of differential equations. Their rather general and pretty theory (Prelle & Singer, 1983) unifies a number of previous results. One example of the kind of equations to which their results apply are systems of first-order, non-linear, ordinary differential equations. In their 1985 paper they apply their results to two-dimensional autonomous systems of ordinary differential equations of the form

$$\begin{cases} \dot{x}(t) = P(x, y) \\ \dot{y}(t) = Q(x, y), \end{cases}$$

where P and Q are polynomials. They reduce the problem of obtaining a decision procedure for determining elementary first integrals for such systems to that of bounding degrees of algebraic solutions of such systems. The latter is a difficult, open problem, but their procedure may prove to be useful in practice by guessing at bounds since solutions of high degree are not likely to be of interest. See Schwarz (1985) for a discussion of applications and a REDUCE implementation of some related ideas.

A key to progress in both integration in finite terms and the solution of differential equations in closed form has been the development of an algebraic setting and theory for these problems. A fundamental supporting result has been the so-called structure theorem, first enunciated by Risch (1969*b*) and given further treatment by Ax (1971), Risch (1979), Rosenlicht (1976), and Rothstein & Caviness (1979). The Rosenlicht paper is a particular favourite of mine.

There have been many papers written on methods for symbolic solutions of differential equations. For example, Della Dora & Tournier (1981) have developed methods for generating formal power series solutions in the neighbourhood of singular points of linear homogeneous differential equations with rational coefficients. Char (1981), Glinos & Saunders (1984) and Watanabe (1981, 1984) have given methods based on mostly classical ideas that work for certain kinds of equations.

2.7. PRODUCTION OF COMPUTER ALGEBRA SYSTEMS

Just as computer algebra would not exist as we know it today without algorithmic improvements for computing gcd's and factoring polynomials, the same can be said about the development of computer algebra systems. There have been dozens of computer algebra systems implemented over the past 20 years. Given the size of modern day computer algebra systems and the sophistication of their algorithms, I believe that they must be regarded as one of our society's technical wonders!

As I think of the computer algebra systems that have had the most impact, four immediately spring to mind: MACSYMA, REDUCE, ALDES/SAC-2 and mu-MATH, each for different reasons. The system with the broadest scope of implemented mathematics is, without a doubt, MACSYMA (Carette & Harten, 1985; Macsyma, 1985). With its relatively recent unleashing from its bondage in a PDP-10 at MIT, its influence is likely to become even more pervasive.

REDUCE is important for several reasons. With its early emphasis on portability, it has been one of the most widely available, comprehensive algebra systems and has been instrumental in bringing computer algebra to many users. Unlike MACSYMA, unless there are some recent MACSYMA implementations of which I am unaware, REDUCE is available on today's super computers such as the CRAY's, CDC CYBER's and the large IBM mainframes. Both MACSYMA and REDUCE are available on individual workstations, particularly those based on the Motorola 68000 family of processor chips.

The ALDES/SAC-2 (Collins & Loos, 1986) system has been highly portable and widely available for over 10 years also, but, in my estimation, this is not the most important characteristic of this system. It is the only substantial system that has had its algorithms carefully and completely documented. Consequently, implementations of other systems are able to incorporate and build on top of a suitably modified version of the SAC-2 "library". As computer algebra software matures, I think that we need to pay more attention to the details of correctness and documentation, otherwise new systems are not able to take sufficient advantage of the advances made by previous systems as they should.

Stoutemyer's (1985) mu-MATH has the potential to be the world's most popular computer algebra system, if it is not already, because it is the most comprehensive algebra system that runs on the IBM PC. To help understand the significance of this, let me tell you about the computing environment at Delaware. We have well over a dozen VAX 11/780/785 machines, an IBM 3081D, a CDC CYBER 174 plus some single user machines that could support MACSYMA or REDUCE. Perhaps we could support 100 simultaneous MACSYMA users, perhaps 200 simultaneous REDUCE or SMP users and maybe 400 MAPLE users. However, of our 800+ faculty members, over 500 of them have IBM PC's or AT's. There are perhaps 200 additional IBM PC's on campus, thus making it possible to support over 700 simultaneous mu-MATH users! This is the significance of mu-MATH. Besides, it is superbly engineered for the small machine environment. However, if systems like MACSYMA, REDUCE, SMP or MAPLE become available on similarly inexpensive machines, as they may, the significance of mu-MATH may not be as great.

Of the newer systems MAPLE (Char *et al.*, 1986), SCRATCHPAD (Sutor, 1985) and SMP (Wolfram *et al.*, 1983) are the best known. Grosheva *et al.* (1983) have compiled an extensive survey of computer algebra systems.

Another important development in the past 5 years or so is the commercialisation of computer algebra software. Gone are the days (if they ever existed) when everyone in the computer algebra community freely made their code available at nominal cost. Today we are into the world of proprietary software and trade secrets. Nonetheless, I think that in the long run commercialisation can be positive, especially if the commercial ventures turn out to be successful. Consider, for example, today's computer industry as a whole. Clearly if computers had remained the sole premise of dusty university and industrial research labs, computing would not have progressed anywhere near as fast as it has since 1950 when the first fledgling computer companies began.

Commercial companies tend to be much better at some important tasks than universities. Examples that come to mind are documentation and software maintenance, activities that, with a few notable exceptions, universities tend to do badly. In general, commercial companies have to be careful about smoothing out the rough edges in software; otherwise they will lose their customers. We all tend to benefit by these activities. At the same time we must do all that we can to insure that important new scientific ideas are freely available to the research community so that the overall health of the field will remain strong.

3. Open Problems and Future Directions

In this section I would like to explore some unsolved problems whose solution I believe to be particularly worthwhile. There are computational problems in all areas of mathematics. As mathematics marches on, an unending line of computer algebra

problems arise. Once I heard someone say that all the significant problems in computer algebra had been solved. This is like saying that all the significant problems in mathematics have been solved!

3.1. SOFTWARE

Despite the progress that has been made to date in software for algebraic computation, the software area remains one with substantial problems. As a researcher on algebraic algorithms, there is still no computer algebra system that is satisfactory for my purposes. For the implementation of new procedures for integration in finite terms or for solving differential equations in closed form, I must have a system with supporting facilities in this area. This requirement alone eliminates all but a few systems. Of those that have implemented algorithms for transcendental function arithmetic, it is next to impossible to find out the details of the specification of the algorithms, much less the details of the algorithms themselves. One is usually reduced to reading hundreds of lines of Lisp code, an unsatisfactory state of affairs. Research on new algorithms is far outstripping our implementation capabilities. Many state of the art algorithms have been implemented in only one or two systems, if at all!

There is some promise on the horizon. The work at IBM Yorktown Heights Research Center on new SCRATCHPAD suggests that we may be able to obtain an order of magnitude reduction in the size of the written code for algorithms specified in these languages. The compaction is accomplished through ideas of abstraction applied to algorithm descriptions. Abstraction is a common theme in programming language design today, but it should be a particularly fruitful tool when applied to a well-structured application area like algebraic computation. If one can reduce 100 pages of Lisp code to ten pages of SCRATCHPAD code, as seems to be possible, then that will make it substantially easier to produce new computer algebra software. Yet there remain unresolved questions about the utility of this new approach. Hopefully, the availability of SCRATCHPAD over Arpanet and CSNet will help to answer some of the questions, help to focus attention on problems of this approach, and lead to algebraic software systems that will greatly enhance our overall productivity.

My second unsolved problem is really a proposal to solve a problem. Our colleagues in numerical computation have been successful over the past 10 years in improving the quality and robustness of numerical software by developing carefully crafted and documented libraries of state of the art numerical software. Notable examples are the IMSL and NAG libraries.

I believe the time has come to carry out a similar project for algebraic software. Although there would be many questions to resolve before beginning production of such a library, many algorithms could be implemented without substantial new research, a major benefit compared to other possible ways of improving our software environment. Of course, many questions of software engineering, such as languages to be used, questions of portability, organisation of a development team, quality control and funding must be settled before actual code could be produced. The task is large, but the pay-off could be even larger.

3.2. TRANSCENDENTAL FUNCTION ARITHMETIC

There are numerous problems in this area. Perhaps the most important are algorithms for definite integration and definite summation.

In his Ph.D. thesis, Wang (1971) implemented some heuristic methods for definite integration. See also Campbell *et al.* (1976). Fox & Hearn (1974) described a REDUCE program for evaluation of some definite integrals that occur in quantum electrodynamics. Their work extended that of Peterman. More recently, Kölbig has written a series of papers in which he gives a method for the evaluation of a class of definite integrals involving powers of exponentials and logarithms. See Kölbig (1985) for a report on this work and references to his previous papers. Gosper has done some unpublished work on definite summation, but otherwise I know of no work in this area.

Another important problem, perhaps one with far reaching consequences because it is similar to a number of other problems, is the one of determining bounds for the so-called Risch–Norman (or parallel Risch) "algorithm" for integration in finite terms. This problem was first posed at the 1976 SYMSAC meeting in Yorktown Heights. Davenport (1982) has made some progress, but the most difficult parts of the problem remain unsolved.

The Singer, Saunders, Caviness extended Liouville theory contains some important limitations; notably, it does not cover the dilogarithm function. The extension to cover this and similar functions will seemingly require some new techniques and hence may lead to some techniques for further development of the theory.

Cherry's and Knowles' algorithms for integrating in terms of logarithms or error functions are incomplete in several ways. First, one would like to extend the algorithms to more special functions of interest. Second, one would like to be able to express the integral in terms of more than one of these functions, i.e. instead of "algorithms for integrating in terms of error functions *or* logarithmic integrals" we would like "algorithms for integrating in terms of error functions, logarithmic integrals *and* other special functions". All of this needs to be generalised to include algebraic functions in the integral.

Other problems have to do with structure theorems for fields of higher order functions. A particular example is given a linear ode with coefficients in a field k, determine the transcendence degree over k of the differential field obtained by adjoining the solutions of the differential equation. We must have such results before we can do arithmetic in fields containing higher order functions.

A goal of some researchers is to develop decision procedures for determining the solution of differential equations in terms of the solutions of a given set of differential equations. Such algorithms would make closed-form solutions of differential equations much more interesting and useful in practice. Singer (1985) has shown how to solve nth order linear homogeneous differential equations in terms of Liouvillian functions and solutions of second-order linear homogeneous differential equations. This is an interesting result, but not quite what is desired, in that the set of equations that specifies the solution set cannot be specified. In 1984 Davenport gave additional open problems having to do with integration in finite terms.

3.3. POLYNOMIAL IDEALS AND THE THEORY OF REAL CLOSED FIELDS

Despite the inherently exponential nature of these two problem areas, the continued improvement and application of algorithms for these problems should receive increasing attention. The problems are just too important and fundamental to ignore. Work on algorithms for computing Gröbner bases has just scratched the surface of what is possible. There are many aspects of this algorithm that need to be studied.

The successful applications that recently have been reported of Collins' algorithm for the theory of real closed fields to some non-trivial and interesting problems holds forth the promise that computers may yet be useful assistants in proving difficult results. This is an exciting possibility.

3.4. PARALLEL ALGORITHMS AND ARCHITECTURES

Recently the notion of parallelism has been receiving much attention in computer science, especially in the architecture, programming languages, artificial intelligence and theory areas. Complexity classes for parallel computation have been devised by the theory community and the AI and computer architecture researchers have proposed a number of highly parallel machines. It seems likely that parallelism is here to stay. This will be an increasingly important topic within algebraic computation in the coming years.

There have been a few papers dealing with computer algebra issues and parallel computing. Marti & Fitch (1983) report on the design of a general purpose multiprocessor using Motorola 68000 microprocessors for research in symbolic computation and expert systems. Buchberger (1985*c*) discusses a parallel machine project to study large parallel machines of arbitrary interconnection topology for the purposes of exploiting parallelism in symbolic algorithms. Watt (1985) describes a prototype parallel system for running parallel MAPLE computer algebra programs. His prototype parallel system is a local area network of VAX 11/780's.

There are two different approaches to what constitutes a fast parallel algorithm. One is to start with a sequential algorithm and to try to introduce parallelism to obtain an n-fold speed-up on n processors as is attempted on the architectures mentioned above. Sasaki & Kanada (1981) give parallel algebraic algorithms, in this sense, for calculating determinants and solving linear equations.

The second has a theoretical objective to make the parallel time as small as possible, allowing an almost arbitrary (e.g. polynomially bounded) number of processors. There have been several papers that present fast parallel algorithms in this second sense for algebraic problems.

Borodin *et al.* (1982) present parallel algorithms for solving systems of linear equations and computing the gcd of polynomials. von zur Gathen (1984) presents fast parallel solutions to the problems of computing the extended Euclidean scheme (or algorithm) of two polynomials over an arbitrary field, computing the gcd and lcm of many polynomials over an arbitrary field, factoring polynomials over finite fields and computing the square-free decomposition of polynomials over fields of characteristic p, $p \geqslant 0$. Kaltofen *et al.* (1986) give fast parallel algorithms for computing the Hermite and Smith normal form of matrices with polynomial entries.

3.5. EDUCATION

There are two issues in education—education about computer algebra and using computer algebra systems for education. Both of these are old topics. Education about the use of computer algebra systems will continue to increase as these systems become more widely available. Engeler & Mäder (1985) report about an educational program at ETH. The goal of their program is the laudable one of introducing students to the entire range of modern software for solving mathematical problems of which computer algebra is just one topic. Others are undoubtedly starting similar programs. In the Fall of 1985 we taught an applied algebraic computation course at the University of Delaware for the first

time. Text material is a problem. We used the book *Computer Algebra in Applied Mathematics: An Introduction to Macsyma* by R. H. Rand. We supplemented it with problems in applied mathematics and additional material on MACSYMA. The course was team taught by Robert Gilbert, a colleague in the Department of Mathematical Sciences, and myself.

A goal of the MAPLE system is to provide a computer algebra system that can be used to teach large numbers of students. Apparently this is beginning to happen at the University of Waterloo.

Education about algebraic algorithms is another important area. The new text by Buchberger *et al.* (1983) is a welcome addition. But a definitive text is still not available. Writing such a text is an important, but demanding, undertaking. Perhaps attention should be restricted to particular subsets of computer algebra thereby making the task less formidable. On the other hand, I have some concern that the education of computer scientists, at least in the US, is headed in a direction that will make it more difficult to get good students to study computer algebra. A major problem is a lack of sufficient mathematical backgrounds for many computer science students.

At Johannes Kepler University in Linz a comprehensive curriculum in symbolic computation for students in both computer science and mathematics was adopted in 1982 (Buchberger, 1984). This curriculum includes computer algebra, computational logic and automatic programming. It is surely one of the most extensive programs offered in this area by any university.

The promise of the use of computer algebra software as an educational tool in mathematics is still largely unfulfilled as is the application of computer algebra software to computer aided instruction. The new emphasis on AI techniques in CAI may help to bring the use of computer algebra more to the forefront in this area.

In August 1984, at the Fifth International Congress on Mathematical Education held in Adelaide, Australia, a group of about thirty persons worked in four meetings on symbolic mathematical systems and their effects on the curriculum. The report of this group distinguishes three broad areas in which computer algebra will affect education: (i) courses about computer algebra, (ii) courses on how to use computer algebra, and (iii) pedagogical uses of computer algebra. The double issue (Vol. 18, No. 4, Nov. 1984 and Vol. 19, No. 1, Feb. 1985) of the *SIGSAM Bulletin* contains the papers that were presented on these three topics.

4. Conclusion

The preceding just touches on some of the highlights of the accomplishments and unsolved problems in computer algebra. A really comprehensive survey would be much longer. I close with the following quote from the 1870 address of James C. Maxwell to the Mathematics and Physics Section of the British Association for the Advancement of Science, which perhaps helps to keep a proper perspective on our work: "For the sake of persons of ... different types, scientific truth should be presented in different forms, and should be regarded as equally scientific, whether it appears in the robust form and the vivid colouring of a physical illustration, or in the *tenuity and paleness of a symbolic expression* (my emphasis)."

Preliminary copies of this paper were shared with a number of colleagues. For their suggestions I am appreciative. Erich Kaltofen's detailed comments were particularly helpful.

References

Arnon, D. S. (1981). *Algorithms for the geometry of semi-algebraic sets.* Ph.D. Thesis, Tech. Report No. 436, Computer Sciences Dept., Univ. of Wisconsin.

Arnon, D. S. (1983). Topologically reliable display of algebraic curves. *Proc. of SIGGRAPH '83,* pp. 219–227. New York: ACM.

Arnon, D. S. (1985). On mechanical quantifier elimination for elementary algebra and geometry: solution of a nontrivial problem. Proc. of Eurocal '85, Vol. II (Caviness, B. F., ed.), *Springer Lec. Notes Comp. Sci.* **204,** 271.

Arnon, D. S., Collins, G. E., McCallum, S. (1984). Cylindrical algebraic decomposition I: the basic algorithm. *SIAM J. Comp.* **13,** 865–877.

Ax, J. (1971). On Schanuel's conjectures. *Ann. Math.* **93,** 252–268.

Bell, E. T. (1937). *Men of Mathematics.* New York: Simon and Schuster.

Berlekamp, E. R. (1967). Factoring polynomials over finite fields. *Bell System Tech. J.* **46,** 1853–1859.

Böge, W. (1980). Decision procedures and quantifier elimination for elementary real algebra and parametric polynomial nonlinear optimization. Preliminary manuscript.

Borodin, A., von zur Gathen, J., Hopcroft, J. (1982). Fast parallel matrix and gcd computations. *Inf. Control* **52,** 241–256.

Brown, W. S. (1971). On Euclid's algorithm and the computation of polynomial greatest common divisors. *J. Assoc. Comp. Mach.* **18,** 478–504.

Brown, W. S. (1976). On the subresultant PRS algorithm. In: (Jenks, R. D., ed.) *Proc. 1976 ACM Symposium on Symbolic and Algebraic Computation,* p. 271. New York: ACM.

Brown, W. S. (1978). The subresultant PRS algorithm. *ACM TOMS* **4,** 237–249.

Brown, W. S., Hyde, J. P., Tague B. A. (1964). The ALPAK system for non-numerical algebra on a digital computer II: rational functions of several variables and truncated power series with rational function coefficients. *Bell System Tech. J.* **43,** 785–804.

Brown, W. S., Traub, J. F. (1971). On Euclid's algorithm and the theory of subresultants. *J. Assoc. Comp. Mach.* **18,** 505–514.

Buchberger, B. (1965). An algorithm for finding a basis for the residue class ring of a zero-dimensional polynomial ideal (German), Ph.D. Thesis, Math. Inst., Univ. of Innsbruck, Austria and *Aequ. Math.* **4,** (1970) 374–383.

Buchberger, B. (1976*a*). A theoretical basis for the reduction of polynomials to canonical form. *ACM SIGSAM Bull.* **10,** 3, 19–29.

Buchberger, B. (1976*b*). Some properties of Gröbner bases for polynomial ideas. *ACM SIGSAM Bull.* **10,** 4, 19–24.

Buchberger, B. (1979). A criterion for detecting unnecessary reductions in the construction of Gröbner bases. In: (Ng, E. W., ed.) Proc. 1979 European Symposium on Symbolic and Algebraic Manipulation. *Springer Lec. Notes Comp. Sci.* **72,** 3–21.

Buchberger, B. (1983*a*). A note on the complexity of constructing Gröbner bases. In: (van Hulzen, J. A., ed.) Computer Algebra: Proceedings of Eurocal '83. *Springer Lec. Notes Comp. Sci.* **162,** 137–145.

Buchberger, B. (1983*b*). A critical-pair/completion algorithm for finitely generated ideals in rings. In: Proc. of the Conf. Rekursive Kombinatorik. *Springer Lec. Notes Comp. Sci.* **171,** 137–161.

Buchberger, B. (1984). CAMP: a teaching project in symbolic computation at the University of Linz. *SIGSAM Bull.* **18,** 4, 8–9.

Buchberger, B. (1985*a*). Gröbner bases: an algorithmic method in polynomial ideal theory. In: (Bose, N. K., ed.) *Multidimensional Systems Theory,* pp. 184–232. Dordrecht: Reidel.

Buchberger, B. (1985*b*). The parallel L-machine for symbolic computation. In: (Caviness, B. F., ed.) Proc. of Eurocal '85, Vol. II. *Springer Lec. Notes Comp. Sci.* **204,** 541–542.

Buchberger, B., Collins, G. E., Loos, R. (eds) (1983). *Computer Algebra: Symbolic and Algebraic Computation,* 2nd edn. Vienna: Springer-Verlag.

Buchberger, B., Loos, R. (1983). Algebraic simplification. In: (Buchberger, B., Collins, G. E., Loos, R., eds) *Computer Algebra: Symbolic and Algebraic Computation,* 2nd edn, pp. 11–43. Vienna: Springer-Verlag.

Campbell, J. A., Kent, J. G., Moore, R. J. (1976). Experiments with a symbolic programming system for complex analysis. *BIT* **16,** 241–256.

Carrette, G. J., Harten, L. P. (1985). DOE-Macsyma: progress report. Proc. of Eurocal '85, Vol. II (Caviness, B. F., ed.). *Springer Lec. Notes Comp. Sci.* **204,** 36–39.

Char, B. W. (1981). Using Lie transformation groups to find closed form solutions to first order ordinary differential equations. In: (Wang, P. S., eds) *Proc. 1981 ACM Symp. on Symbolic and Algebraic Computation,* pp. 44–50. New York: ACM.

Char, B. W., Fee, G. J., Geddes, K. O., Gonnet, G. H., Monagan, M. B. (1986). A tutorial introduction to Maple. *J. Symbolic Computation* **2,** 179–200.

Cherry, G. W. (1983). *Algorithms for integrating elementary functions in terms of logarithmic integrals and error functions.* Ph.D. Thesis, Univ. of Delaware.

Cherry, G. W. (1985). Integration in finite terms with special functions: the error function. *J. Symbolic Computation* **1**, 283–302.
Cherry, G. W. (1986). Integration in finite terms with special functions: the logarithmic integral. *SIAM J. Comput.* **15**, 1–21.
Chistov, A. L., Grigor'ev, D. Yu. (1982). Polynomial-time factoring of the multivariate polynomials over a global field. LOMI preprint E-5-82, Leningrad.
Chistov, A. L., Grigor'ev, D. Yu. (1983*a*). Subexponential-time solving systems of algebraic equations I. LOMI preprint E-9-83, Leningrad.
Chistov, A. L., Grigor'ev, D. Yu. (1983*b*). Subexponential-time solving systems of algebraic equations II. LOMI preprint E-10-83, Leningrad.
Chistov, A. L., Grigor'ev, D. Yu. (1984). Complexity of quantifier elimination in the theory of algebraically closed fields. In: Proc. of the 11th Symp. on Math. Found. of Comp. Sci. (Chytil, M. P., Koubek, V., eds). *Springer Lec. Notes Comp. Sci.* **176**, 17–31.
Cohen, P. J. (1969). Decision procedures for real and p-adic fields. *Comm. Pure Appl. Math.* **22**, 131–151.
Collins, G. E. (1966). PM, a system for polynomial manipulation. *Commun. ACM* **9**, 578–589.
Collins, G. E. (1967). Subresultants and reduced polynomial remainder sequences. *J. Assoc. Comp. Mach.* **14**, 128–142.
Collins, G. E. (1972). *The SAC-2 polynomial GCD and resultant system.* Univ. of Wisconsin Comp. Sci. Tech. Report No. 145.
Collins, G. E. (1974). The computing time of the Euclidean algorithm. *SIAM J. Comput.* **3**, 1–10.
Collins, G. E. (1975). Quantifier elimination for real closed fields by cylindrical algebraic decomposition. Proc. of Second GI Conf. on Automata Theory and Formal Languages. *Springer Lec. Notes Comp. Sci.* **33**, 134–183.
Collins, G. E. (1979). Factoring univariate integral polynomials in polynomial average time. In: Proc. 1979 European Symposium on Symbolic and Algebraic Manipulation (Ng, E. W., ed.). *Springer Lec. Notes Comp. Sci.* **72**, 317–329.
Collins, G. E. (1983). Quantifier elimination for real closed fields: a guide to the literature. In: (Buchberger, B., Collins, G. E., Loos, R., eds) *Computer Algebra: Symbolic and Algebraic Computation*, 2nd edn, pp. 79–81. Vienna: Springer-Verlag.
Collins, G. E., Akritas, A. G. (1976). Polynomial real root isolation using Descartes' rule of signs. In: (Jenks, R. D., ed.) *Proc. 1976 ACM Symposium on Symbolic and Algebraic Computation*, pp. 272–275. New York: ACM.
Collins, G. E., Loos, R. (1983). Real zeros of polynomials. In: (Buchberger, B., Collins, G. E., Loos, R., eds) *Computer Algebra: Symbolic and Algebraic Computation*, 2nd edn, pp. 83–94. Vienna: Springer-Verlag.
Collins, G. E., Loos, R. (1986). *ALDES/SAC-2 Reference Manual.* Vienna: Springer-Verlag.
Davenport, J. H. (1981). On the Integration of Algebraic Functions. *Springer Lec. Notes Comp. Sci.* **102**.
Davenport, J. H. (1982). The parallel Risch algorithm. In: Computer Algebra: Proceedings of Eurocam '82 (Calmet, J., ed.). *Springer Lec. Notes Comp. Sci.* **144**, 144–157.
Davenport, J. H. (1984). Integration in finite terms. *SIGSAM Bull.* **18**, 2, 20–21.
Della Dora, J., Tournier (1981). Formal solutions of differential equations in the neighborhood of singular points. In: (Wang, P. S., ed.) *Proc. 1981 ACM Symp. on Symbolic and Algebraic Computation*, pp. 25–29. New York: ACM.
Dixon, J. D. (1970). The number of steps in the Euclidean algorithm. *J. Number Theor.* **2**, 414–422.
Engeler, E., Mäder, R. (1985). Scientific computation: the integration of symbolic, numeric and graphic computation. Proc. of Eurocal '85, Vol. I (Buchberger, B., ed.). *Springer Lec. Notes Comp. Sci.* **203**, 185–200.
Fischer, M. J., Rabin, M. O. (1974). Super exponential complexity of Presburger arithmetic. *Complexity of Computation*, SIAMS-AMS Proc., Vol. VII, Amer. Math. Soc., 27–41.
Floyd R. W. (ed.) (1966). Proceedings of the ACM symposium on symbolic and algebraic manipulation. *Commun. ACM* **9**, 547–643.
Fox, J. A., Hearn, A. C. (1974). Analytic computation of some integrals in fourth order quantum electrodynamics. *J. Comput. Phys.* **14**, 301–317.
Gebauer, R., Kredel, H. (1984). An algorithm for constructing Gröbner bases of polynomial ideals. *SIGSAM Bull.* **18**, 9.
Glinos, N., Saunders, B. D. (1984). Operational calculus techniques for solving differential equations. Proc. 1984 European Symp. on Symbolic and Algebraic Manipulation (Fitch, J., ed.). *Springer Lec. Notes Comp. Sci.* **174**, 23–34.
Gosper, R. W. (1978). Decision procedure for indefinite hypergeometric summation. *Proc. Nat. Acad. Sci.* **75**, 1, 40–42.
Grosheva, M. V. *et al.* (1983). Computer algebra systems on computers (Analytical packages of application programs), preprint, Keldysh Inst. Appl. Math., USSR Academy of Sciences, Tech. Report No. 1.
Harrington, S. J. (1979). A new symbolic integration system for Reduce. *Comput. J.* **22**, 127–131.
Heilbronn, H. A. (1969). On the average length of a class of finite continued fractions. In: (Turán, P., ed.) *Number Theory and Analysis*, pp. 87–96. New York: Plenum Press.

Heindel, L. E. (1971). Integer arithmetic algorithms for polynomial real zero determination. *J. Assoc. Comp. Mach.* **18,** 533–548.

Hermann, G. (1926). The question of finitely many steps in polynomial ideal theory (German). *Math. Annal.* **95,** 736–788.

Hironaka, H. (1964). Resolution of singularities of an algebraic variety over a field of characteric zero: I, II. *Ann. Math.* **79,** 109–326.

Huet, G., Oppen, D. C. (1980). Equations and rewrite rules. In: (Book, R. V., ed.) *Formal Language Theory: Perspectives and Open Problems*, pp. 349–405. New York: Academic Press.

Johnson, S. C. (1966). *Tricks for improving Kronecker's polynomial factoring algorithm.* Bell Telephone Labs Tech. Report.

Jordon, D. E., Kain, R. Y., Clapp, L. C. (1966). Symbolic factoring of polynomials in several variables. *Commun. ACM* **8,** 638–643.

Kaltofen, E. L. (1982*a*). Factorization of polynomials. In: (Buchberger, B., Collins, G. E., Loos, R., eds) *Computer Algebra: Symbolic and Algebraic Computation*, 2nd edn, pp. 95–113. Vienna: Springer-Verlag.

Kaltofen, E. L. (1982*b*). *On the complexity of factoring polynomials with integer coefficients.* Ph.D. Thesis, Rensselaer Polytechnic Inst.

Kaltofen, E. L. (1982*c*). A polynomial-time reduction from bivariate to univariate integral polynomial factorization. *Proc. 23rd Symposium on Foundations of Computer Science*, pp. 57–64. IEEE.

Kaltofen, E. L. (1982*d*). A polynomial reduction from multivariate to bivariate polynomial factorization. *Proc. 14th Annual ACM Symp. on the Theory of Comput.*, pp. 261–266.

Kaltofen, E. L. (1985). Polynomial-time reductions from multivariate to bi- and univariate integral polynomial factorization. *SIAM J. Comput.* **14,** 469–489.

Kaltofen, E. L., Krishnamoorthy, M., Saunders, B. D. (1986). Fast parallel computation of Hermite and Smith forms of polynomial matrices. *SIAM J. Alg. and Discrete Methods* (in press).

Kaltofen, E. L., Musser, D. R., Saunders, B. D. (1983). A generalized class of polynomials that are hard to factor. *SIAM J. Comput.* **12,** 473–483.

Kandri-Rody, A., Kapur, D. (1983). On the relationship between Buchberger's Gröbner basis algorithm and the Knuth–Bendix completion procedure. TIS Report No. 83CRD286, General Electric R & D Ctr.

Kandri-Rody, A., Kapur, D. (1984*a*). Algorithms for computing Gröbner bases of polynomial ideals over various Euclidean rings. Proc. 1984 European Symp. on Symbolic and Algebraic Manipulation (Fitch, J., ed.). *Springer Lec. Notes Comp. Sci.* **174,** 195–206.

Kandri-Rody, A., Kapur, D. (1984*b*). Computing the Gröbner basis of polynomial ideals over integers. *Proc. of 3rd Macsyma Users' Conf.* Schenectady, New York.

Karr, M. (1981). Summation in finite terms. *J. Assoc. Comp. Mach.* **28,** 305–350.

Karr, M. (1985). Theory of summation in finite terms. *J. Symbolic Computation* **1,** 303–315.

Knowles, P. (1986). *Symbolic integration in terms of error functions and logarithmic integrals.* Ph.D. Thesis, North Carolina State Univ.

Knuth, D. E. (1969). *The Art of Computer Programming.* Vol. 2: *Seminumerical Algorithms.* Reading, Mass.: Addison-Wesley.

Kölbig, K. S. (1985). Explicit evaluation of certain definite integrals involving powers of logarithms. *J. Symbolic Computation* **1,** 109–114.

Kovacic, J. J. (1979). An algorithm for solving second order linear homogeneous differential equations, manuscript. Also in *J. Symbolic Computation* **2,** (1986) 3–43.

Landau, S. (1985). Factoring polynomials over algebraic number fields. *SIAM J. Comput.* **14,** 184–195.

Landau, S., Miller, G. (1983). Solvability by radicals is in polynomial time. *Proc. 15th Annual ACM Symp. on Theory of Comp.*, pp. 140–151.

Lauer, M. (1976). Canonical representatives for residue classes of a polynomial idea. In: (Jenks, R. D., ed.) *Proc. 1976 ACM Symposium on Symbolic and Algebraic Computation*, pp. 339–345. New York: ACM.

Lauer, M. (1982). Computing in homomorphic images. In: (Buchberger, B., Collins, G. E., Loos, R., eds) *Computer Algebra: Symbolic and Algebraic Computation*, 2nd edn, pp. 139–168. Vienna: Springer-Verlag.

Lazard, D. (1981). Résolutions des systèmes d'équations algébriques. *Theor. Comp. Sci.* **15,** 77–110.

Lazard, D. (1983). Gröbner bases, Gaussian elimination and resolution of systems of algebraic equations. In: (van Hulzen, J. A., ed.) Computer Algebra: Proceedings of Eurocal '83. *Springer Lec. Notes Comp. Sci.* **162,** 146–156.

Lenstra, A. K. (1983). Factoring polynomials over algebraic number fields. In: (van Hulzen, J. A., ed.) Computer Algebra: Proceedings of Eurocal '83. *Springer Lec. Notes Comp. Sci.* **162,** 245–254.

Lenstra, A. K. (1984). Factorization of polynomials. *SIGSAM Bull.* **18,** 2, 16–17.

Lenstra, A. K., Lenstra, H. W., Lovász, L. (1982). Factoring polynomials with rational coefficients. *Math. Annal.* **261,** 515–534.

Loos, R. (1974). Toward a formal implementation of computer algebra. Proc. 1974 European Symposium on Symbolic and Algebraic Manipulation. *ACM SIGSAM Bull.* **8,** 3, 9–16.

Loos, R. (1982). Generalized polynomial remainder sequences. In: (Buchberger, B., Collins, G. E., Loos, R., eds) *Computer Algebra: Symbolic and Algebraic Computation*, 2nd edn, pp. 115–137. Vienna: Springer-Verlag.

Macsyma (1985). *Vax Unix Macsyma Reference Manual.* Version 11, Symbolics, Inc.
Marti, J., Fitch, J. (1983). The Bath concurrent Lisp machine. In: (van Hulzen, J. A., ed.) Computer Algebra: Proceedings of Eurocal '83. *Springer Lec. Notes Comp. Sci.* **162,** 78–85.
Mayr, E. W., Meyer, A. R. (1982). The complexity of the word problems for commutative semigroups and polynomial ideals. *Adv. Math.* **46,** 305–329.
Mora, F., Möller, H. M. (1983). The computation of the Hilbert function. In: (van Hulzen, J. A., ed.) Computer Algebra: Proceedings of Eurocal '83. *Springer Lec. Notes Comp. Sci.* **162,** 157–167.
Moses, J. (1967). *Symbolic integration.* Ph.D. Thesis, M.I.T., also Project MAC Tech. Report TR-47.
Moses, J. (1969). The integration of a class of special functions with the Risch algorithm. *SIGSAM Bull.* **13,** 13–27.
Moses, J., Yun, D. Y. Y. (1973). The EZGCD algorithm. *Proc. ACM National Conf.*, pp. 159–166.
Müller, F. (1978). Ein exakter Algorithmus zur nichtlinearen Optimierung für beliebige Polynome mit mehreren Veranderlichen, Verlag Anton Hain, Meisenheim am Glan.
Musser, D. R. (1971). *Algorithms for polynomial factorization.* Ph.D. Thesis, Dept. of Computer Sciences, Univ. of Wisconsin.
Musser, D. R. (1976). Multivariate polynomial factorization. *J. Assoc. Comp. Mach.* **22,** 291–308.
Pan, V. (1984). *On sequential and parallel evaluation of polynomial zeroes and of matrix polynomials.* Tech. Report 84-6, Computer Science Dept., SUNY Albany.
Pinkert, J. R. (1976). An exact method for finding the roots of a complex polynomial. *ACM TOMS* **2,** 351–363.
Pohst, M., Yun, D. Y. Y. (1981). On solving systems of algebraic equations via ideal bases and elimination theory. In: (Wang, P. S., ed.) *Proc. 1981 ACM Symp. on Symbolic and Algebraic Computation,* pp. 206–211. New York: ACM.
Porter, J. W. (1975). On a theorem of Heilbronn. *Mathematika* **22,** 20–28.
Prelle, M. J. (1982). *Elementary first integrals of differential equations.* Ph.D. Thesis, Rensselaer Polytechnic Institute.
Prelle, M. J., Singer, M. F. (1983). Elementary first integrals of differential equations. *Trans. Amer. Math. Soc.* **279,** 215–229.
Richardson, D. (1968). Some unsolvable problems involving elementary functions of a real variable. *J. Symb. Logic* **33,** 511–520.
Risch, R. H. (1968). *On the integration of elementary functions which are built up using algebraic operations.* System Development Corp. Tech. Report SP-2801/002/00.
Risch, R. H. (1969*a*). The problem of integration in finite terms. *Trans. Amer. Math. Soc.* **139,** 167–189.
Risch, R. H. (1969*b*). *Further results on elementary functions.* IBM Research Tech Report RC 2402.
Risch, R. H. (1970). The solution of the problem of integration in finite terms. *Bull. Amer. Math. Soc.* **76,** 605–608.
Risch, R. H. (1979). Algebraic properties of the elementary functions of analysis. *Amer. J. Math.* **101,** 743–759.
Rolletschek, H. (1983). The Euclidean algorithm for Gaussian integers. In: (van Hulzen, J. A., ed.) Computer Algebra: Proceedings of Eurocal '83. *Springer Lec. Notes Comp. Sci.* **162,** 12–23.
Rosenlicht, M. (1976). On Liouville's theory of elementary functions. *Pacific J. Math.* **65,** 485–492.
Rothstein, M. (1976). *Aspects of symbolic integration and simplification of exponential and primitive functions.* Ph.D. Thesis, Unov. of Wisconsin.
Rothstein, M., Caviness, B. F. (1979). A structure theorem for exponential and primitive functions. *SIAM J. Comput.* **8,** 357–367.
Sasaki, T., Kanada, Y. (1981). Parallelism in algebraic computation and parallel algorithms for symbolic linear systems. In: (Wang, P. S., ed.) *Proc. 1981 ACM Symp. on Symbolic and Algebraic Computation,* pp. 160–167. New York: ACM.
Saunders, B. D. (1981). An implementation of Kovacic's algorithm for solving second order linear homogeneous differential equations. In: (Wang, P. S., ed.) *Proc. 1981 ACM Symp. on Symbolic and Algebraic Computation*, pp. 105–108. New York: ACM.
Schaller, S. (1979). *Algorithmic aspects of polynomial residue class rings.* Ph.D. Thesis, Dept. of Comp. Sci., Univ. of Wisconsin.
Schwarz, F. (1985). An algorithm for determining polynomial first integrals of autonomous systems of ordinary differential equations. *J. Symbolic Computation* **1,** 229–233.
Seidenberg, A. (1954). A new decision method for elementary algebra. *Ann. Math.* **60,** 365–374.
Singer, M. F. (1977). Functions satisfying elementary relations. *Trans. Amer. Math. Soc.* **227,** 185–206.
Singer, M. F. (1981). Liouvillian solutions of nth order homogeneous linear differential equations. *Amer. J. Math.* **103,** 661–682.
Singer, M. F. (1985). Solving homogeneous linear differential equations in terms of second order linear differential equations. *Amer. J. Math.* **107,** 663–696.
Singer, M. F., Saunders, B. D., Caviness, B. F. (1985). An extension of Liouville's theorem on integration in finite terms. *SIAM J. Comput.* **14,** 966–990.
Slagle, J. R. (1961). *A heuristic program that solves symbolic integration problems in freshman calculus, symbolic automatic integrator (SAINT).* Ph.D. Thesis, Mass. Inst. of Tech.
Slisenko, A. O. (1981). Complexity problems in computational theory. *Russian Math. Surveys* **30,** 23–125.

Spear, D. (1977). A constructive approach to commutative ring theory. In: (Lewis, V. E., ed.) *Proc. 1977 Macsyma Users' Conf.*, pp. 369–376. Cambridge, Mass.: MIT.

Stoutemyer, D. R. (1985). A preview of the next IBM-PC version of muMATH. In: (Buchberger, B., ed.) Proc. of Eurocal '85, Vol. I. *Springer Lec. Notes Comp. Sci.* **203,** 33–44.

Sutor, R. S. (1985). The Scratchpad II computer algebra language and system. In: (Caviness, B. F., ed.) Proc. of Eurocal '85, Vol. II. *Springer Lec. Notes Comp. Sci.* **204,** 32–33.

Szekeres, G. (1952). A canonical basis for the ideals of a polynomial domain. *Am. Math. Monthly* **59,** 279–386.

Tarski, A. (1948). *A decision method for elementary algebra and geometry,* 2nd revised edn. Univ. of California Press.

Trager, B. M. (1976). Algebraic factoring and rational function integration. In: (Jenks, R. D., ed.) *Proc. 1976 ACM Symposium on Symbolic and Algebraic Computation,* pp. 219–226. New York: ACM.

Trager, B. M. (1979). Integration of simple radical extensions. In: (Ng, E. W., ed.) Proc. 1979 European Symposium on Symbolic and Algebraic Manipulation. *Springer Lec. Notes Comp. Sci.* **72,** 408–414.

van der Waerden, B. L. (1953). *Modern Algebra,* Vol. I. New York: Frederick Ungar.

von zur Gathen, J. (1983). Factoring sparse multivariate polynomials. *Proc. of the 24th Annual IEEE Symp. on the Foun. of Comp. Sci.*, pp. 172–179.

von zur Gathen, J. (1984). Parallel algorithms for algebraic problems. *SIAM J. Comput.* **13,** 802–824.

Wang, P. S. (1971). *Evaluation of definite integrals by symbolic manipulation.* MIT Project MAC TR-92.

Wang, P. S. (1977). Preserving sparseness in multivariate polynomial factorization. In: (Lewis, V. E., ed.) *Macsyma Users' Conf.*, pp. 55–61. Cambridge, Mass.: MIT.

Wang, P. S. (1978). An improved multivariate polynomial factoring algorithm. *Math. Comp.* **32,** 1215–1231.

Wang, P. S. (1979). Analysis of the p-adic construction of multivariate correction coefficients in polynomial factorization: iteration vs recursion. In: (Ng, E. W., ed.) Proc. 1979 European Symposium on Symbolic and Algebraic Manipulation. *Springer Lec. Notes Comp. Sci.* **72,** 291–300.

Wang, P. S., Rothschild, L. P. (1975). Factoring multivariate polynomials over the integers. *Math. Comp.* **29,** 935–950.

Watanabe, S. (1981). A technique for solving ordinary differential equations using Riemann's P-function. In: (Wang, P. S., ed.) *Proc. 1981 ACM Symp. on Symbolic and Algebraic Computation,* pp. 36–43. New York: ACM.

Watanabe, S. (1984). An experiment toward a general quadrature for second order linear ordinary differential equations by symbolic computation. In: (Fitch, J., ed.) Proc. 1984 European Symp. on Symbolic and Algebraic Manipulation. *Springer Lec. Notes Comp. Sci.* **174,** 13–22.

Watt, S. M. (1985). A system for parallel computer algebra programs. In: (Caviness, B. F., ed.) Proc. of Eurocal '85, Vol. II. *Springer Lec. Notes Comp. Sci.* **204,** 537–538.

Williams, L. H. (1962). Algebra of polynomials in several variables for a digital computer. *J. Assoc. Comp. Mach.* **9,** 29–40.

Winkler, F. (1983). An algorithm for constructing detaching bases in the ring of polynomials over a field. In: (van Hulzen, J. A., ed.) Computer Algebra: Proceedings of Eurocal '83. *Springer Lec. Notes Comp. Sci.* **162,** 168–179.

Winkler, F. (1984). On the complexity of the Gröbner-basis algorithm over K[x, y, z]. In: (Fitch, J., ed.) European Symp. on Symbolic and Algebraic Manipulation. *Springer Lec. Notes Comp. Sci.* **174,** 184–194.

Winkler, F., Buchberger, B., Lichtenberger, F., Rolletschek, H. (1985). An algorithm for constructing canonical bases of polynomial ideals. *ACM TOMS* **11,** 66–78.

Wolfram, S. *et al.* (1983). *SMP Reference Manual.* Los Angeles, CA: Inference Corp.

Yun, D. Y. Y. (1974). *The Hensel lemma in algebraic manipulation.* Ph.D. Thesis, M.I.T., MAC-TR-138.

Zacharias, G. (1978). *Generalized Gröbner bases in commutative polynomial rings.* Bachelor Thesis, MIT.

Zassenhaus, H. (1969). On Hensel factorization I. *J. Number Theor.* **1,** 287–292.

Zippel, R. E. (1979*a*). *Probabilistic algorithms for sparse polynomials.* Ph.D. Thesis, MIT.

Zippel, R. E. (1979*b*). Probabilistic algorithms for sparse polynomials. In: (Ng, E. W., ed.) Proc. 1979 European Symposium on Symbolic and Algebraic Manipulation. *Springer Lec. Notes Comp. Sci.* **72,** 216–226.